UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS CORNÉLIO PROCÓPIO
DIRETORIA DE GRADUAÇÃO
DEPARTAMENTO ACADÊMICO DE ENGENHARIA ELÉTRICA
ENGENHARIA DE CONTROLE E AUTOMAÇÃO

NATÁLIA NOGARA STÁBILE

CONTROLE PID BASEADO EM NEURÔNIO ARTIFICIAL

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
2021

NATÁLIA NOGARA STÁBILE

CONTROLE PID BASEADO EM NEURÔNIO ARTIFICIAL

PID control based on artificial neuron

Trabalho de Conclusão de Curso de graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia de Controle e Automação da Universidade Tecnológica Federal do Paraná (UTFPR).
Orientador: Prof. Dr. Kleber Romero Felizardo

CORNÉLIO PROCÓPIO

2021

Universidade Tecnológica Federal do Paraná
Campus Cornélio Procópio
Departamento Acadêmico de Elétrica
Curso de Engenharia de Controle e Automação

FOLHA DE APROVAÇÃO

Natália Nogara Stábile

Controle PID baseado em neurônio artificial

Trabalho de conclusão de curso apresentado às 18:00hs do dia 14/05/2021 como requisito parcial para a obtenção do título de Engenheiro de Controle e Automação no programa de Graduação em Engenharia de Controle e Automação da Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Avaliadora composta pelos professores abaixo assinados. Após deliberação, a Banca Avaliadora considerou o trabalho aprovado.

_____

Prof(a). Dr(a). Kleber Romero Felizardo - Presidente (Orientador)

_____

Prof(a). Dr(a). Edson Aparecido Rozas Theodoro - (Membro)

_____

Prof(a). Dr(a). Luiz Francisco Sanches Buzachero - (Membro)

A folha de aprovação assinada encontra-se na coordenação do curso.

Dedico este trabalho à minha mãe Marinês e ao meu pai Edenilson, que com muito esforço, carinho e dedicação fizeram com que meu sonho de cursar Engenharia fosse possível.

AGRADECIMENTOS

Primeiramente gostaria de agradecer à Deus pela vida e pela força dada a mim para que eu pudesse ultrapassar todos os obstáculos encontrados ao longo da minha jornada e durante todo o tempo de realização deste trabalho.

Também gostaria de agradecer a minha família por todo o amor, apoio e incentivo dado ao longo de todos esses anos, em especial aos meus pais que nunca pouparam esforços para realizar meus sonhos e me fazerem feliz, e ao meu irmão Mateus, pela cumplicidade e por sempre me animar com seu jeito risonho e amigável de ser.

Agradeço a todos os professores, desde os da pré-escola até os da graduação, por me mostrarem os caminhos do conhecimento e me incentivarem a continuar estudando, mas principalmente ao meu professor e orientador Kleber, por seu enorme conhecimento e paciência, pela imensa ajuda ao longo do desenvolvimento do trabalho e por acreditar que esse projeto fosse possível.

Não posso deixar de agradecer a todos os meus amigos pelo companheirismo, pelos momentos de alegria que vivenciamos e por estarem comigo também nas situações de dificuldade. Por fim, agradeço a todos que sempre torceram por mim e que de alguma forma contribuíram para a realização deste trabalho.

"É melhor lançar-se à luta em busca do triunfo, mesmo expondo-se ao insucesso, que formar fila com os pobres de espírito, que nem gozam muito, nem sofrem muito; e vivem nessa penumbra cinzenta sem conhecer vitória e nem derrota."

Franklin D. Roosevelt

RESUMO

STÁBILE, Natália Nogara. **Controle PID baseado em neurônio artificial**. 2021. 43 f. Trabalho de Conclusão de Curso – Engenharia de Controle e Automação, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2021.

Este trabalho de conclusão de curso apresenta a simulação do controle de dois sistemas lineares, sendo um de primeira e outro de segunda ordem e o controle de um sistema não linear de segunda ordem, utilizando um controlador PID. Os parâmetros do PID foram sintonizados por meio dos pesos sinápticos de uma rede neural artificial composta por um único neurônio. Essa escolha se deve à simplicidade e boa performance dos controladores PID e à facilidade computacional e de manipulação de não linearidades que tal rede propicia. Para a simulação do controlador, foi utilizado um algoritmo composto pela Regra de Hebb supervisionada e pela Regra de Hebb não supervisionada. Os resultados das simulações mostram que o controlador PID neural apresentou um ótimo desempenho para os três sistemas simulados.

Palavras-chave: PID neural. Controle neural. Sistemas não lineares. Redes neurais artificiais. Controle digital.

ABSTRACT

STÁBILE, Natália Nogara. **PID control based on artificial neuron**. 2021. 43 f. Course conclusion paper – Control and Automation Engineering, Federal University of Technology - Paraná. Cornélio Procópio, 2021.

This course conclusion paper presents the simulation of the control of two linear systems, being one of first order and the other of second order, and the control of a nonlinear system of second order, using a PID controller. The PID parameters are tuned using the synaptic weights of an artificial neural network composed of a single neuron. This choice is due to the simplicity and good performance of the PID controllers and the computational and nonlinearity manipulation facilities that this network provides. For the controller simulation, an algorithm composed of the supervised Hebb Rule and the unsupervised Hebb Rule was used. The results of the simulations show that the neural PID controller presented an excellent performance for the three simulated systems.

Keywords: PID neural. Neural control. Nonlinear systems. Artificial neural networks. Digital control.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Uma das formas mais usuais de controlar um sistema é utilizando um controlador PID, sendo que o mesmo representa mais da metade dos controladores empregados na indústria atualmente (OGATA, 2010). Tal controlador confere um controle satisfatório e é aplicado a maioria dos sistemas. Sua ampla aplicação pode ser atribuída à sua simplicidade e à sua boa performance em uma larga faixa de condições de operação (DORF; BISHOP, 2001).

Entretanto, a maioria das técnicas clássicas de projeto de controladores PID fornecem apenas uma estimativa inicial para seus parâmetros, sendo que frequentemente os mesmos são sintonizados empiricamente (DONG; HU; REN, 2008). Uma outra limitação do PID convencional é que o mesmo não costuma ser eficiente em sistemas complexos e com imprecisões, sistemas não lineares e em sistemas no qual não há um modelo matemático preciso (FALLAHI; AZADI, 2009).

No caso dos sistemas não lineares, além das estratégias clássicas de controle, muitas pesquisas estão sendo desenvolvidas utilizando técnicas de sistemas inteligentes para controlá-los. Segundo (BEHDAD JAMSHIDI; JAMSHIDI; ROSTAMI, 2017), seu uso fornece soluções mais rápidas e precisas para analisar sistemas não lineares em comparação aos métodos convencionais. Tais sistemas simulam características de sistemas biológicos para implementar hardwares e softwares que tenham a capacidade de aprender e se adaptar (FLOARES, 2013).

As redes neurais artificiais (RNAs) são algumas das técnicas computacionais inseridas no conceito de sistemas inteligentes. Essas técnicas são baseadas no sistema nervoso de seres vivos e são compostas por neurônios artificiais que se intercomunicam por meio de conexões que simulam a sinapse humana (SILVA; SPATTI; FLAUZINO, 2010). As mesmas possuem as características de adaptabilidade, tolerância a falhas e capacidade de aprendizado e generalização (HAYKIN, 2001).

Neste trabalho é realizada a simulação do controle de dois sistemas lineares, no qual um é de primeira ordem e outro de segunda e o controle de um sistema não linear de segunda ordem, utilizando um PID neural, cujos parâmetros são determinados utilizando uma RNA com um único neurônio.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é projetar e utilizar um controlador PID neural, cujos parâmetros são sintonizados por meio de uma rede composta por um único neurônio, para controlar dois sistemas lineares, sendo um de primeira e outro de segunda ordem e um sistema não linear de segunda ordem.

1.1.2 Objetivos Específicos

- Aplicar o algoritmo de treinamento da rede para determinar os parâmetros do PID;
- Simular o controle dos sistemas utilizando o PID neural;
- Analisar os resultados obtidos do controlador PID neural aplicados aos sistemas simulados.

1.2 JUSTIFICATIVA

Sistemas reais possuem não linearidades como atrito de Coulomb, atrito viscoso e atrito de Stribeck. Algumas dessas não linearidades são descontínuas e comumente observadas tanto em pequenos intervalos de operação quanto em grandes (SLOTINE; LI, 1991). Tais não linearidades não permitem uma aproximação linear local e, portanto, sistemas que apresentam esse tipo de característica necessitam da utilização de técnicas de controle não linear, dependendo do intervalo de operação desejado.

Uma das formas de controlar sistemas não lineares é utilizando redes neurais artificiais, uma vez que as mesmas possuem a capacidade de modelar tais sistemas (ZILKOVA; JAROSLAV; GIROVSKY, 2006). Nesse contexto, a RNA de neurônio único mostra-se como uma alternativa para o desenvolvimento de controladores neurais devida à sua simplicidade estrutural e facilidade computacional (SHENG et al., 2002). Por outro lado, os controladores PID apresentam uma formulação matemática simples e um bom desempenho. Assim, a união da simplicidade do PID com a habilidade das redes neurais de lidarem com não linearidades torna-se útil no controle de sistemas não lineares.

2 REVISÃO BIBLIOGRÁFICA

A utilização das redes neurais artificiais para o controle de sistemas foi apresentada, pela primeira vez, por (WERBOS, 1989) e por (NARENDRA; PARTHASARATHY, 1990). No primeiro trabalho foi proposto um controlador neural em malha fechada e, no segundo, um controle adaptativo neural. A partir disso, vários trabalhos utilizando RNAs para controlar sistemas foram desenvolvidos, cujas aplicações se estendem a diversas áreas.

Na área industrial, foi proposto um controlador adaptativo aplicado a um sistema do tipo batelada de fermentação utilizando RNAs (BARUCH et al., 2003). Na robótica, (JIN, 1993) desenvolveu um controle neural para a trajetória de um braço robótico. Já no setor aeroespacial, (XU et al., 2013) elaboraram um controlador com RNA aplicado à dinâmica longitudinal de uma aeronave hipersônica.

Os primeiros trabalhos utilizando o controle PID neural surgiram no início de 90, dentre eles, um controlador para um manipulador robótico (TOKITA et al., 1991) e um controle aplicado a sistemas não lineares utilizando a rede Perceptron multicamadas (TAN; HE, 1992). Também foram desenvolvidos o controle de velocidade de um veículo elétrico (MATSUMURA; OMATU; HIGASA, 1993) e um controlador direto usando redes convolucionais (BAHRAMI; TAIT, 1993).

Devida à sua simplicidade, os algoritmos baseados em redes com um único neurônio também passaram a ser utilizados na implementação de controladores PID neurais. Alguns exemplos disso são a utilização de um controlador para conversores CC (WANG; XU; DING, 2008), o controle de um veículo aéreo não tripulado (MOBAREZ; SARHAN; ASHRY, 2019) e um controlador aplicado a um sistema de microfermentação não linear, variante no tempo e com atraso temporal (WANG; LI, 2013).

2.1 REDES NEURAIS ARTIFICIAIS

Uma RNA pode ser definida da seguinte forma:

> Uma rede neural é um processador maciçamente e paralelamente distribuído, constituído de unidades de processamento simples, que têm a propensão natural de armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:
>
> 1. O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem.
>
> 2. Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido. (HAYKIN, 2001).

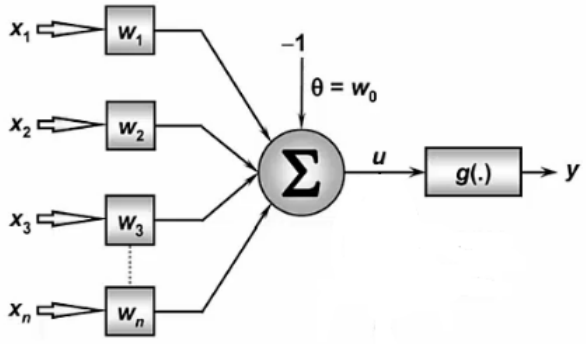
As redes neurais possuem algumas características que as tornam úteis em diversas aplicações. São elas (SILVA; SPATTI; FLAUZINO, 2010):

- Habilidade de se adaptar por experiência: uma vez que o ajuste dos pesos sinápticos é realizado por meio de uma série de apresentações de exemplos de padrões relacionados ao sistema em questão, a rede possui a característica de aprender por exemplos;

- Capacidade de aprendizado: por meio da utilização de um algoritmo de treinamento, a rede é capaz de extrair a interconexão existente entre as diferentes variáveis que envolvem o sistema;
- Habilidade de generalização: após o treinamento da rede, a mesma possui a habilidade de generalizar o aprendizado obtido para encontrar as soluções de padrões que não foram apresentados à rede no processo de treinamento;
- Organização de dados: a rede é capaz de organizar informações recebidas a fim de agrupar características em comum;
- Tolerância a falhas: no caso da rede sofrer um dando interno suave, seu resultado sofrerá apenas uma leve degradação ao invés de uma falha severa;
- Armazenamento distribuído: o aprendizado sobre o comportamento de um sistema é efetuado de forma distribuída entre as inúmeras sinapses de seus neurônios, aumentando assim a robutez da rede;
- Facilidade de prototipagem: de forma geral, a maioria das arquiteturas de RNA pode ser facilmente prototipada em softwares ou hardwares.

McCulloch e Pitts desenvolveram em 1943 a primeira pesquisa na área. Nesse trabalho foi proposto o primeiro modelo matemático baseado em um neurônio biológico, dando origem, ao conceito de neurônio artificial (SILVA; SPATTI; FLAUZINO, 2010). Tal modelo é mostrado na Figura 1.

Figura 1 – Modelo matemático de um neurônio



Fonte: (SILVA; SPATTI; FLAUZINO, 2010)

Nessa representação, o neurônio é constituído da seguinte forma:

As entradas $\{x_1, x_2, ..., x_n\}$ são as medidas provenientes do ambiente externo. Os pesos sinápticos $\{w_1, w_2, ..., w_n\}$ são os valores que fazem a ponderação entre as entradas da rede. O combinador linear $\sum$ tem a função de realizar o somatório ponderado das entradas.

O bias θ ou limiar de ativação é uma variável que define o valor que faz com que o resultado do combinador linear seja propagado para a saída do neurônio. O potencial de ativação p é a diferença entre o resultado do combinador linear e o bias. A função de ativação $g(.)$ é uma função matemática que tem como objetivo limitar a amplitude da saída da rede para melhor adequá-la ao problema tratado. O sinal y é a saída do neurônio.

A partir desse modelo, a saída do neurônio é dada por:

$$y = g(p) \tag{1}$$

No qual:

$$p = \sum_{i=1}^{n} w_i x_i - \theta \tag{2}$$

2.1.1 Paradigmas de Aprendizado

A capacidade de aprender a partir de seu ambiente e de aprimorar sua performance por meio da aprendizagem é umas das características mais importantes de uma rede neural artificial. Isso ocorre por meio do treinamento, que é um processo iterativo de correções que são aplicadas aos pesos da rede até se obter o resultado desejado (HAYKIN, 2001).

O treinamento da RNA segue um algoritmo de aprendizagem, que pode ser definido como um conjunto de passos ordenados necessários para determinar os valores dos pesos sinápticos que geram a saída esperada. Há uma variedade de algoritmos de aprendizagem, cuja diferença está na forma como os pesos são calculados.

A forma como a rede neural se relaciona com o ambiente também é um elemento importante para seu aprendizado e é denominado paradigma de aprendizagem (HAYKIN, 2001). Dois paradigmas clássicos são: aprendizagem supervisionada e aprendizagem não supervisionada.

Na aprendizagem supervisionada ou aprendizagem com um professor, a rede tem disponível cada amostra de entrada com seu respectivo valor de saída. (SILVA; SPATTI; FLAUZINO, 2010). Dessa forma, é como se houvesse um "professor", com conhecimento sobre o ambiente, que mostrasse para a rede a resposta desejada para cada amostra de treinamento.

Na aprendizagem não supervisionada ou auto-organizada, não há um agente externo indicando as saídas desejadas. Desse modo, a rede deve se auto-organizar, com base nas interconexões existentes entre cada um dos elementos do conjunto de amostras, a fim de identificar similaridades e, consequentemente, chegar a resposta desejada (SILVA; SPATTI; FLAUZINO, 2010).

2.2 PID NEURAL

Os controladores PID (Proporcial, Integral e Derivativo) são controladores cujo sinal de controle $u(t)$ é composto por três parcelas: proporção do erro, integral do erro e derivada do

erro. Sua formulação matemática no domínio do tempo é mostrada na Equação 3, no qual K_p é denominado de ganho proporcional, T_i de tempo integrativo e T_d de tempo derivativo.

$$u(t) = K_p[e(t) + \frac{1}{Ti}\int_0^t e(t)dt + T_d\frac{de}{dt}] \tag{3}$$

A parcela proporcional tem a função de diminuir os erros de regime permanente e distúrbios, porém, não elimina completamente o erro. Além disso, valores elevados de K_p podem aumentar ruídos e levar o sistema à instabilidade (OGATA, 2010).

A ação integral reduz ou até mesmo elimina o erro estacionário, contudo, pode aumentar o valor do sobressinal da resposta do sistema (OGATA, 2010).

O controle derivativo melhora a resposta transitória, entretanto, não pode ser utilizado separadamente, uma vez que o mesmo atua sobre a taxa de variação do erro e não sobre o erro em si (OGATA, 2010).

Dessa forma, o controle PID tem a missão de usar o melhor de cada uma das três ações de controle e, ao mesmo tempo, reduzir ao máximo possível suas desvantagens. Para isso, os parâmetros devem ser sintonizados adequadamente.

Para utilizar o controlador PID em sistemas digitais, como computadores e microcontroladores, o mesmo deve ser utilizado em sua forma discreta. Um dos métodos de transformação do controlador PID contínuo da Equação 3 no controlador discreto é utilizando as aproximações das Equações 4 e 5, no qual T é o tempo de amostragem.

$$\frac{de(t)}{dt} \approx \frac{e(kT) - e(kT - T)}{T} \tag{4}$$

$$\int_0^t e(t)dt \approx \sum_{k=1}^{n} Te(kT) \tag{5}$$

Dessa forma, é possível obter a equação de diferenças do PID discreto, que é dado pela Equação 6 (IBRAHIM, 2006).

$$u(kT) = u(kT-T)+K_p[e(kT)-e(kT-T)]+\frac{K_pT}{T_i}e(kT)+\frac{K_pT_d}{T}[e(kT)-2e(kT-T)-e(kT-2T)] \tag{6}$$

Por simplificação de notação, fica implícito que $u(kT) = u(k)$, $u(kT - T) = u(k - 1)$ e assim por diante. Deste modo, a Equação 6 pode ser reescrita como:

$$u(k) = u(k-1) + K_p[e(k) - e(k-1)] + \frac{K_pT}{T_i}e(k) + \frac{K_pT_d}{T}[e(k) - 2e(k-1) - e(k-2)] \tag{7}$$

Considerando $\frac{K_pT}{T_i} = K_I$ e $\frac{K_pT_d}{T} = K_d$, onde K_I é denominado de ganho integral e K_d, de ganho derivativo, a Equação 7 pode ser reescrita da seguinte forma:

$$u(k) = u(k-1) + K_p[e(k) - e(k-1)] + K_I e(k) + K_d[e(k) - 2e(k-1) - e(k-2)] \quad (8)$$

Um dos algoritmos que pode ser utilizado na aprendizagem do PID de neurônio único, é um algoritmo híbrido, utilizado por (RIVERA-MEJÍA; LEÓN-RUBIO; ARZABALA-CONTRERAS, 2012) e (GÓMEZ-AVILA et al., 2018), composto pela Regra de Hebb supervisionada e pela Regra de Hebb não supervisionada.

A Regra de Hebb supervisionada é mostrada na Equação 9, onde o erro $e(k)$, é a diferença entre a referência e a saída da planta, e atua como um sinal de supervisão, indicando a rede o quão próximo a saída está da referência.

$$w_i(k+1) = w_i(k) + \eta_i e(k) x_i(k) \quad (9)$$

A Regra de Hebb não supervisionada é apresentada na Equação 10 e, ao invés de levar em consideração o erro $e(k)$, utiliza o sinal de controle $u(k)$ para atualizar os pesos.

$$w_i(k+1) = w_i(k) + \eta_i u(k) x_i(k) \quad (10)$$

Unindo as Regras de Hebb supervisionada e não supervisionada, tem-se o algoritmo híbrido citado acima, que é dado pela Equação 11, e que utiliza tanto o erro $e(k)$, quanto a ação de controle $u(k)$, para incrementar os pesos $w_1(k)$, $w_2(k)$ e $w_3(k)$. Essa união faz com que o neurônio seja capaz de auto-organizar as informações presentes no ambiente, sob a supervisão do sinal de erro, e obter o sinal de controle que produz a saída desejada (NING; SHUQUING; ZHANG, 2001).

$$w_i(k+1) = w_i(k) + \eta_i u(k) e(k) x_i(k) \quad (11)$$

O sinal de controle $u(k)$ é dado pela equação 12, no qual $\Phi(I)$ é a função de ativação, onde o argumento I é dado pela Equação 13. Já a constante A_{max}, é a amplitude máxima que o sinal de controle pode atingir.

$$u(k) = u(k-1) + A_{max}\Phi(I) \quad (12)$$

$$I = \sum_{i=1}^{3} x_i(k) w_i(k) \quad (13)$$

A Equação 8 pode ser reescrita conforme mostrada na Equação 14, no qual x_1, x_2 e x_3 representam, respectivamente, o erro proporcional, o erro integral e o erro derivativo. A formulação matemática de cada um deles é apresentada pelo conjunto de Equações 15.

$$u(k) = u(k-1) + K_p x_1(k) + K_I x_2(k) + K_d x_3(k) \tag{14}$$

$$\begin{aligned} x_1(k) &= e(k) - e(k-1) \\ x_2(k) &= e(k) \\ x_3(k) &= e(k) - 2e(k-1) - e(k-2) \end{aligned} \tag{15}$$

Desse modo, é possível notar que:

$$\begin{aligned} K_p(k) &= w_1(k) \\ K_I(k) &= w_2(k) \\ K_d(k) &= w_3(k) \end{aligned} \tag{16}$$

Durante o processo de controle online do PID neural, os novos valores para os pesos $w_1(k)$, $w_2(k)$ e $w_3(k)$ precisam ser normalizados para garantir a convergência da estratégia de controle (XIWEN, 2008), e, são obtidos por meio da Equação 17, no qual $w_i'(k)$ é o novo valor para $w_i(k)$ e é dado pela Equação 18 e Δw_i é o fator de incremento dos pesos, cuja expressão matemática é mostrada pela Equação 19.

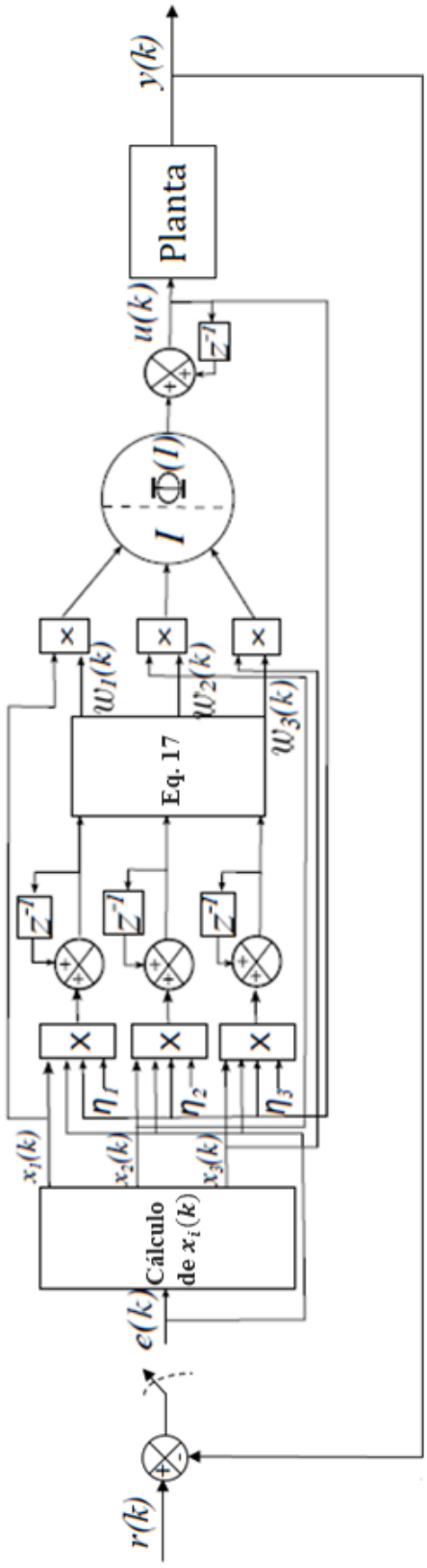
$$w_i(k) = \frac{{w_i}'(k)}{\left\| \sum_{i=1}^{3} {w_i}'(k) \right\|} \tag{17}$$

$${w_i}'(k) = w_i(k-1) + \Delta w_i(k) \tag{18}$$

$$\Delta w_i(k) = \eta_i x_i(k) e(k) u(k) \tag{19}$$

O diagrama de blocos do controle PID neural, no qual são mostradas as interconexões existentes entre cada um dos elementos que compõem o controlador, é apresentado na Figura 2.

Figura 2 – Diagrama de blocos PID Neural



Fonte: (GÓMEZ-AVILA et al., 2018) - Adaptado

3 METODOLOGIA

O processo de desenvolvimento do controlador PID neural pode ser dividido em duas etapas: modelagem matemática dos sistemas e concepção do PID neural. Cada uma dessas etapas é apresentada no decorrer deste capítulo.

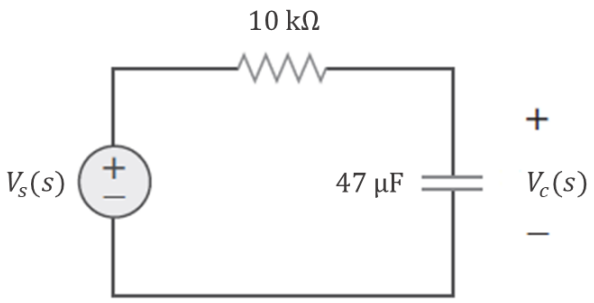
3.1 MODELAGEM MATEMÁTICA DOS SISTEMAS

Para validar o algoritmo desenvolvido, primeiramente foi realizada a simulação em um sistema linear de primeira ordem, em seguida, foi utilizada uma planta também linear, mas de segunda ordem, e, por fim, o PID neural foi aplicado a uma planta não linear. A descrição de cada um dos sistemas utilizados para a validação do algoritmo e seus respectivos modelos matemáticos são mostrados a seguir.

3.1.1 Circuito RC

Para a simulação do algoritmo em uma planta linear de primeira ordem foi utilizado um sistema composto por um resistor e um capacitor em série, assim como mostrado na Figura 3.

Figura 3 – Circuito RC



Fonte: (ALEXANDER; SADIKU, 2013) - Adaptado

A relação entre a saída $Y(s)$, que a é tensão $V_C(s)$ no capacitor, e a entrada $U(s)$, que representa o valor da fonte de tensão $V_S(s)$, é expressa pela função de transferência que é mostrada na Equação 20.

$$\frac{Y(s)}{U(s)} = \frac{1}{0,47s + 1} \tag{20}$$

Para facilitar a simulação dos diferentes sistemas, foi utilizada nesse trabalho a modelagem em tempo discreto. Para isso, foi necessário discretizar a Equação 20 por meio do software MATLAB, com o auxílio da função *c2d*. Tal operação foi feita usando um tempo de amostragem de $0,1s$ e considerando a aproximação pelo segurador de ordem zero (zoh - zero-order hold). O resultado da discretização é apresentado na Equação 21.

$$\frac{Y(z)}{U(z)} = \frac{0,1917}{z - 0,8083} \tag{21}$$

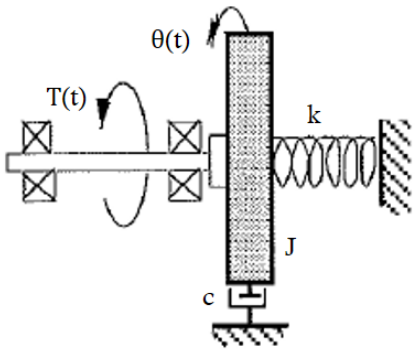
Calculando a transformada Z inversa da função de transferência discreta, foi possível obter a equação de diferenças do sistema, que é mostrada na Equação 22.

$$y(k) = 0,1917u(k-1) + 0,8083y(k-1) \tag{22}$$

3.1.2 Sistema Torcional

Para a simulação do PID neural em uma planta de segunda ordem foi utilizado o modelo de uma planta torcional da empresa ECP, que possui um disco e uma mola, assim como mostrado na Figura 4, no qual J é o momento de inércia do disco, c é o atrito viscoso e k é a constante elástica da mola.

Figura 4 – Sistema torcional



Fonte: (ECP, 1991) - Adaptado

O modelo matemático da planta é dado pela Equação 23, sendo a entrada $U(s)$, o torque $T(s)$ e a saída $Y(s)$, o ângulo $\theta(s)$. Sua função de transferência numérica é mostrada na Equação 24 (ECP, 1991).

$$\frac{Y(s)}{U(s)} = \frac{k}{Jks^2 + cks} \tag{23}$$

$$\frac{Y(s)}{U(s)} = \frac{1600}{s^2 + 0,83s} \tag{24}$$

Para obter a equação de diferenças do sistema torcional, expressa pela Equação 25, foram realizados os mesmos procedimentos efetuados no circuito RC, porém utilizando um tempo de amostragem de $9ms$, uma vez que esse sistema possui uma dinâmica mais rápida do que o circuito RC.

$$y(k) = 0,06464u(k-1) + 0,06448u(k-2) + 1,993y(k-1) - 0,9926y(k-2) \tag{25}$$

3.1.3 Sistema de Controle de Nível

Para a simulação do PID neural em um sistema não linear, foi utilizado o modelo identificado, descrito por (SALES; BILLINGS, 1989), de um sistema de nível de líquido desenvolvido em laboratório. A planta é composta por uma bomba d água de corrente contínua que alimenta um recipiente cônico, que por sua vez, alimenta um tanque de formato quadrado. Sua expressão matemática é dada pela Equação 26, no qual a entrada $u(k)$ representa a tensão da bomba e a saída $y(k)$, a altura da água no recipiente.

$$\begin{aligned} y(k) =& 0,9722y(k-1) + 0,3578u(k-1) - 0,1295u(k-2) \\ &+ 0,3084y(k-1)y(k-2)u(k-2) - 0,04228((y(k-2))^2) \\ &+ 0,1663y(k-2)u(k-2) - 0,03259((y(k-1))^2)y(k-2) \\ &+ 0,3513((y(k-1))^2)u(k-2) - 0,3103y(k-1)u(k-1) \\ &+ 0,1087y(k-2)u(k-1)u(k-2) \end{aligned} \tag{26}$$

3.2 CONCEPÇÃO DO PID NEURAL

Para o desenvolvimento do controle PID neural, foi utilizado o algoritmo híbrido mostrado no capítulo anterior, sendo que a sintonia do controlador foi feita de forma online, ou seja, os parâmetros K_p, K_I e K_d foram aplicados a planta durante o próprio processo de controle. A descrição dos passos realizados no desenvolvimento do controle PID neural é mostrada no Algoritmo 1.

Algoritmo 1 – Algoritmo de controle do PID neural online

Início
1. Especificar os valores r, η_1, η_2, η_3, T, A_{max} e o número de iterações N;
2. Iniciar u, y, e, x_1, x_2, x_3, ϕ, I, Δw_1, Δw_2, Δw_3, ${w_1}'$, ${w_2}'$ e ${w_3}'$ como vetores nulos de tamanho N;
3. Determinar os valores iniciais de w_1, w_2 e w_3;
4. Repetir as instruções $N - 2$ vezes:
4.1 Calcular o erro de regime permanente e;
4.2 Calcular x_1, x_2 e x_3 por meio do conjunto de Equações 15;
4.3 Calcular u utilizando a Equação 12;
4.4 Calcular os pesos atuais w_1, w_2 e w_3 por meio da Equação 17;
4.5 Calcular o novo sinal de controle u usando a Equação 12.
Fim

Fonte: Autoria própria

No código desenvolvido, foi utilizado como sinal de referência, para os três sistemas simulados, um degrau unitário. Como critério de parada da simulação, foi estabelecido o número de iterações, uma vez que é difícil utilizar o erro de regime permanente como método de parada,

já que na sintonia de um controlador PID também são levados em conta outros critérios de desempenho, como a resposta de regime transitório, por exemplo.

As taxas de aprendizagem η_1, η_2 e η_3, os valores iniciais de w_1, w_2 e w_3 e o valor A_{max} foram definidos empiricamente, após alguns testes, enquanto que os valores de erro para as duas primeiras iterações foram definidos como zero, uma vez que o controle PID possui dois atrasos no tempo. Por esse motivo, as instruções foram repetidas $N - 2$ vezes.

Nesse algoritmo foi desconsiderado o limiar de ativação pelo fato do mesmo não possuir uma representação física no contexto do controle PID. Como função de ativação, foi escolhida a tangente hiperbólica, cuja fórmula é mostrada na Equação 27, por ser uma função amplamente utilizada na área de controle de sistemas.

$$\phi(I) = \frac{1 - e^{-I}}{1 + e^{-I}} \tag{27}$$

Os códigos-fonte das simulações do PID neural aplicado a cada um dos sistemas foram desenvolvidos no software Matlab e podem ser vistos com mais detalhes no Apêndice A.

4 RESULTADOS E DISCUSSÕES

Neste capítulo são apresentados os resultados da simulação do PID Neural aplicado a cada um dos sistemas descritos anteriormente, bem como a análise dos mesmos.

4.1 CONTROLE APLICADO AO CIRCUITO RC

Após várias simulações, foram selecionados os valores de η_1, η_2, η_3, A_{max} e os valores iniciais de w_1, w_2 e w_3. Esses valores são mostrados na Tabela 1.

Tabela 1 – Parâmetros da simulação do circuito RC

Parâmetro	Valor
η_1	$0,1$
η_2	$0,1$
η_3	$0,01$
$w_1(0)$	$1,5$
$w_2(0)$	$0,1$
$w_3(0)$	$0,0$
A_{max}	5

Fonte: Autoria própria

Os parâmetros do controlador PID obtidos após a simulação são apresentados na Tabela 2.

Tabela 2 – Parâmetros do PID para o circuito RC

Parâmetro	Valor
K_p	$0,9640$
K_I	$0,2656$
K_d	$0,0117$

Fonte: Autoria própria

Na Figura 5 são apresentados os gráficos da saída da planta compensada pelo PID neural, do sinal de controle $u(k)$ e da convergência de cada um dos pesos durante a simulação. Nessa figura também é mostrada a resposta da saída do sistema em malha aberta. Para a simulação do controlador e a plotagem dos gráficos foi adotado um tempo de amostragem T de $0,1s$ e um valor de k que varia de 1 a 100 com incremento de 1.

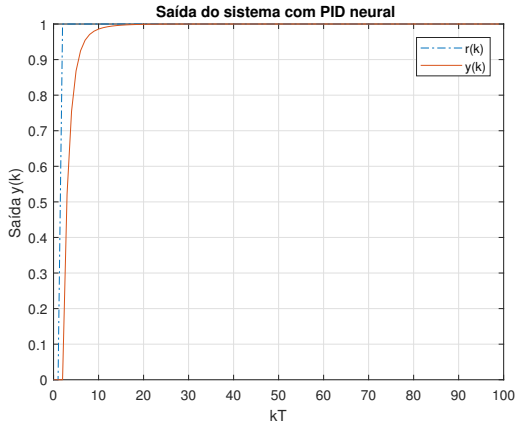
Analisando a Figura 5, é possível notar que a saída da planta após a aplicação do PID seguiu o sinal de referência de forma satisfatória. Além disso, nota-se também que a saída foi estabilizada em um tempo menor em comparação à saída do sistema em malha aberta.

4.2 CONTROLE APLICADO AO SISTEMA TORCIONAL

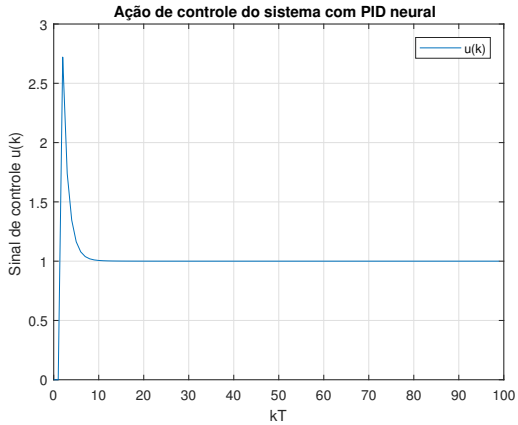
Os valores selecionados para os parâmetros η_1, η_2, η_3, A_{max} e os valores iniciais de w_1, w_2 e w_3 são apresentados na Tabela 3. A amplitude dos ganhos do controlador PID obtidos

Figura 5 – Resultados do PID neural aplicado ao circuito RC

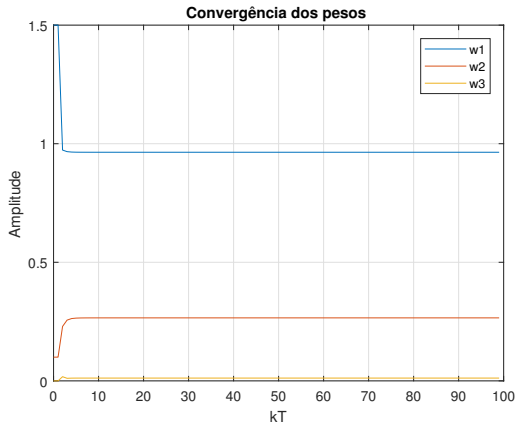
(a) Saída do sistema com PID neural



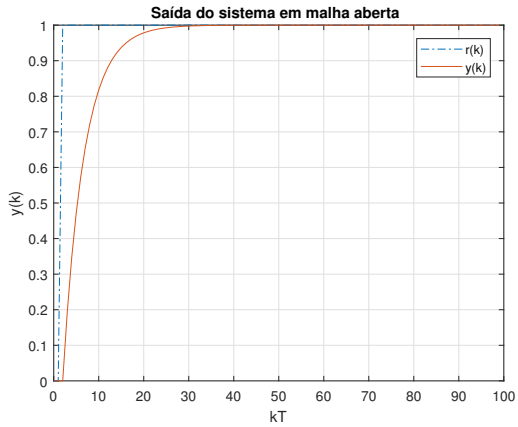
(b) Ação de controle do sistema com PID neural



(c) Convergência dos pesos



(d) Saída do sistema em malha aberta



Fonte: Autoria própria

por meio da simulação é mostrada na Tabela 4.

Tabela 3 – Parâmetros da simulação do sistema torcional

Parâmetro	Valor
η_1	$0,009$
η_2	$0,00001$
η_3	$0,09$
$w_1(0)$	$0,0009$
$w_2(0)$	$0,0001$
$w_3(0)$	$0,01$
A_{max}	5

Fonte: Autoria própria

São mostrados na Figura 6 os gráficos da saída do sistema com o PID Neural, do sinal de controle $u(k)$, da convergência dos pesos durante a simulação e da resposta da saída do sistema em malha aberta, adotando-se um tempo de amostragem T de $9ms$ e um valor de k que varia de 1 a 250 com incremento de 1.

Analisando a Figura 6, observa-se que a saída do sistema após a aplicação do PID

Tabela 4 – Parâmetros do PID para o sistema torcional

Parâmetro	Valor
K_p	$0,1133$
K_I	$0,0102$
K_d	$0,9935$

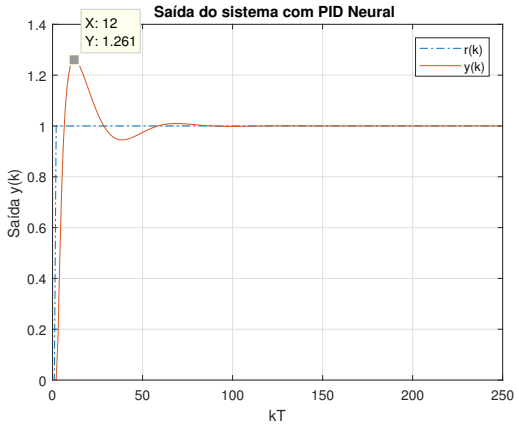
Fonte: Autoria própria

seguiu a referência, obtendo apenas um sobressinal de $26,1\%$.

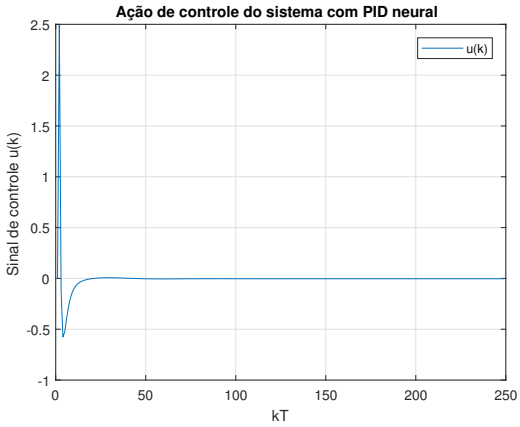
Observando as Tabelas 3 e 4 também é possível notar que a taxa de aprendizagem η_3, o valor inicial de w_3 e o valor final de K_d são maiores se comparados aos demais parâmetros do PID. Isso pode ser explicado pelo fato da parcela derivativa melhorar a resposta transiente, que é o maior problema da dinâmica do sistema ao fechar a malha de controle, uma vez que sistemas torcionais são sujeitos a vibrações de torção que fazem com que o sistema apresente um comportamento transitório com muitas oscilações.

Figura 6 – Resultados do PID neural aplicado ao sistema torcional

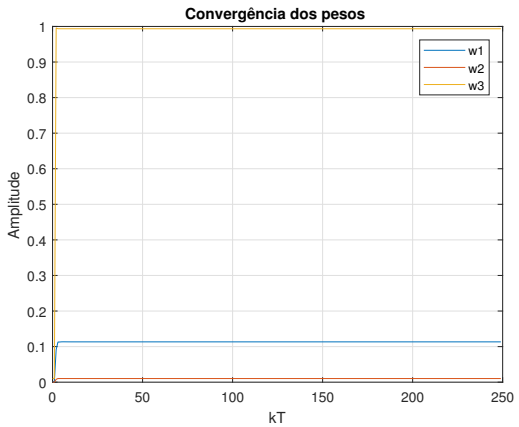
(a) Saída do sistema com PID neural



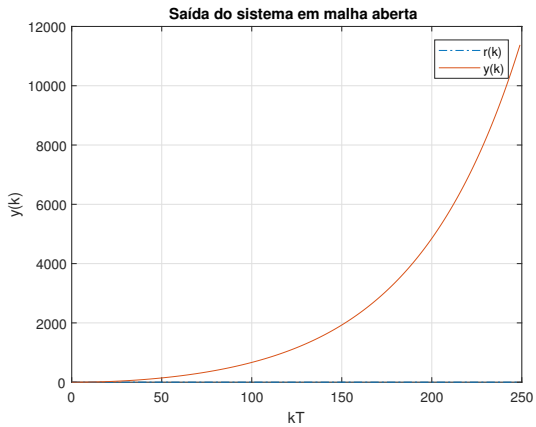
(b) Ação de controle do sistema com PID neural



(c) Convergência dos pesos



(d) Saída do sistema em malha aberta



Fonte: Autoria própria

4.3 CONTROLE APLICADO AO SISTEMA DE CONTROLE DE NÍVEL

Na Tabela 5 são mostrados os valores selecionados para os parâmetros η_1, η_2, η_3, A_{max} e os valores iniciais de w_1, w_2 e w_3. A amplitude dos ganhos do controlador PID obtidos por meio da simulação é apresentada na Tabela 6.

Na Figura 7 são apresentados os gráficos da saída da planta com o PID neural, da ação de controle $u(k)$ e da convergência de cada um dos pesos durante a simulação. Nessa figura também é mostrada a resposta da saída do sistema em malha aberta. Para a simulação do controlador e plotagem dos gráficos foi adotado um tempo de amostragem T de $1s$ e um valor de k que varia de 1 a 250 com incremento de 1.

Tabela 5 – Parâmetros da simulação do sistema de controle de nível

Parâmetro	Valor
η_1	$0,1$
η_2	$0,1$
η_3	$0,1$
$w_1(0)$	$5,7$
$w_2(0)$	$0,1$
$w_3(0)$	$0,1$
A_{max}	1

Fonte: Autoria própria

Tabela 6 – Parâmetros do PID para o sistema de controle de nível

Parâmetro	Valor
K_p	$0,9984$
K_I	$0,0447$
K_d	$0,0334$

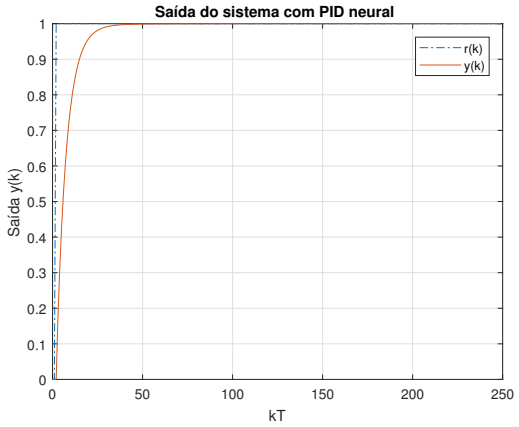
Fonte: Autoria própria

Analisando os resultados, é possível notar que a resposta da saída da planta controlada pelo PID obteve um ótimo desempenho, uma vez seguiu o sinal de referência sem conter sobressinais. Apresentando, dessa forma, uma melhor resposta de saída do que o sistema em malha aberta, que possui um erro estacionário de aproximadamente 1200%.

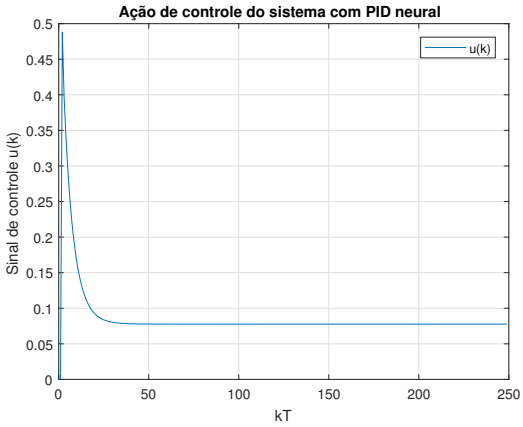
Também nota-se que o valor inicial de w_1 e o valor resultante para o ganho K_p são maiores se comparados com os outros parâmetros. Resultado esse, que tem coerência, uma vez que o controle proporcional tem a função de melhorar o erro de regime permanente.

Figura 7 – Resultados do PID neural aplicado ao sistema de controle de nível

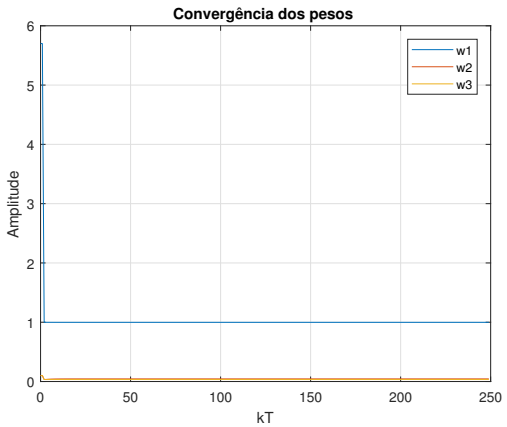
(a) Saída do sistema com PID neural



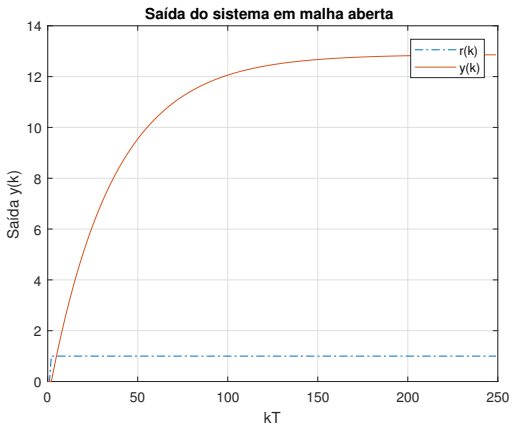
(b) Ação de controle do sistema com PID neural



(c) Convergência dos pesos



(d) Saída do sistema em malha aberta



Fonte: Autoria própria

5 CONCLUSÕES

Tendo em vista a necessidade da utilização da modelagem não linear que alguns sistemas possuem, foi desenvolvido no presente trabalho um controlador PID baseado em um único neurônio artificial, tendo como algoritmo de aprendizagem um algoritmo híbrido, composto pela Regra de Hebb não supervisionada e pela Regra de aprendizagem da rede Perceptron.

O controle foi validado primeiramente em uma planta linear de primeira ordem e em uma planta linear de segunda ordem, antes de ser aplicada em um sistema não linear. Os sistemas escolhidos para a validação do algoritmo apresentam comportamentos distintos entre si, sendo que para a simulação do controlador em uma planta linear de primeira e segunda ordem foram utilizados um circuito RC em série e um sistema torcional, respectivamente. Já na simulação em uma planta não linear, usou-se um sistema de controle de nível de líquido.

Ao analisar os resultados obtidos, verifica-se que o controle desenvolvido apresentou um ótimo desempenho para os três sistemas simulados, sendo que isso fica mais evidente na simulação do sistema torcional e da planta de controle de nível, já que são sistemas que possuem um comportamento menos satisfatório em malha aberta se comparados com o circuito RC.

Diante dos bons resultados alcançados, torna-se significativo o aprofundamento do projeto a fim de melhorar a pesquisa desenvolvida. Dessa forma, sugere-se como possível trabalho futuro, a aplicação da técnica em plantas práticas, com o intuito de averiguar se os resultados experimentais estão de acordo com os simulados. Sugere-se também expor os sistemas simulados a distúrbios, a fim de avaliar melhor o desempenho do controle elaborado.

REFERÊNCIAS

ALEXANDER, Charles.K.; SADIKU, Matthew.N.O. **Fundamentos de Circuitos Elétricos**. 5. ed. [S.l.]: AMGH, 2013. Citado na página 20.

BAHRAMI, M.; TAIT, K. E. Design of direct controllers of pid type by receptive field neural networks. In: **Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)**. [S.l.: s.n.], 1993. v. 2, p. 1809–1812 vol.2. Citado na página 13.

BARUCH, I. S. et al. An adaptive neural control of a fed-batch fermentation processes. In: **Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003.** [S.l.: s.n.], 2003. v. 2, p. 808–812 vol.2. Citado na página 13.

BEHDAD JAMSHIDI, M.; JAMSHIDI, M.; ROSTAMI, S. An intelligent approach for nonlinear system identification of a li-ion battery. In: **2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS)**. [S.l.: s.n.], 2017. p. 98–103. Citado na página 11.

DONG, A.; HU, Q.; REN, Q. The simulation study of adaline pid control and decoupling for the vav system. In: **2008 4th International Conference on Wireless Communications, Networking and Mobile Computing**. [S.l.: s.n.], 2008. p. 1–4. ISSN 2161-9646. Citado na página 11.

DORF, Richard C.; BISHOP, Robert H. **Sistemas de Controle Modernos**. 8. ed. São Paulo: Ltc, 2001. Citado na página 11.

ECP. **Manual for Model 205/205a Torsional Control System**. Bell Canyon: [s.n.], 1991. Citado na página 21.

FALLAHI; AZADI. Adaptive control of a dc motor using neural network sliding mode control. **Lecture Notes in Engineering and Computer Science**, v. 2175, 03 2009. Citado na página 11.

FLOARES, Alexandru. **Computational Intelligence**. New York: Nova Science Publishers, 2013. Citado na página 11.

GÓMEZ-AVILA, J. et al. Control of quadrotor using a neural network based pid. In: **2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI)**. [S.l.: s.n.], 2018. p. 1–6. Citado 2 vezes nas páginas 17 e 19.

HAYKIN, Simon. **Redes neurais : princípios e prática**. 2. ed. Porto Alegre: Bookman, 2001. Citado 3 vezes nas páginas 11, 13 e 15.

IBRAHIM, Dogan. **Microcontroller Applied Digital Control**. Chichester: John Wiley, 2006. Citado na página 16.

JIN, B. Robotic manipulator trajectory control using neural networks. In: **Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)**. [S.l.: s.n.], 1993. v. 2, p. 1793–1796 vol.2. Citado na página 13.

MATSUMURA, S.; OMATU, S.; HIGASA, H. Speed control of an electric vehicle system using pid type neurocontroller. In: **Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)**. [S.l.: s.n.], 1993. v. 1, p. 661–664 vol.1. Citado na página 13.

MOBAREZ, E. N.; SARHAN, A.; ASHRY, M. Fractional order pid based on a single artificial neural network algorithm for fixed wing uavs. In: **2019 15th International Computer Engineering Conference (ICENCO)**. [S.l.: s.n.], 2019. p. 1–7. Citado na página 13.

NARENDRA, K. S.; PARTHASARATHY, K. Identification and control of dynamical systems using neural networks. **IEEE Transactions on Neural Networks**, v. 1, n. 1, p. 4–27, March 1990. ISSN 1045-9227. Citado na página 13.

NING, W.; SHUQUING, W.; ZHANG, J. A model reference intelligent controller using a single neuron. In: **2001 International Conferences on Info-Tech and Info-Net. Proceedings (Cat. No.01EX479)**. [S.l.: s.n.], 2001. v. 4, p. 170–173 vol.4. Citado na página 17.

OGATA, Katsuhiko. **Engenharia de Controle Moderno**. 5. ed. São Paulo: Pearson Prentice Hall, 2010. Citado 2 vezes nas páginas 11 e 16.

RIVERA-MEJÍA, J.; LEÓN-RUBIO, A.G.; ARZABALA-CONTRERAS, E. PID Based on a Single Artificial Neural Network Algorithm for Intelligent Sensors. **Journal of applied research and technology**, scielomx, v. 10, p. 262 – 282, 04 2012. ISSN 1665-6423. Citado na página 17.

SALES, K.R.; BILLINGS, S.A. **Self-Tuning Control of Nonlinear Armax Models**. 1989 — University of Sheffield, 1989. Citado na página 22.

SHENG, Q. et al. Design and implementation of an adaptive pid controller using single neuron learning algorithm. In: **Proceedings of the 4th World Congress on Intelligent Control and Automation (Cat. No.02EX527)**. [S.l.: s.n.], 2002. v. 3, p. 2279–2283 vol.3. Citado na página 12.

SILVA, Ivan N. da; SPATTI, Danilo H.; FLAUZINO, Rogerio A. **Redes neurais artificiais: para engenharia e ciências aplicadas**. São Paulo: Artibiliber, 2010. Citado 4 vezes nas páginas 11, 13, 14 e 15.

SLOTINE, Jean-Jacques E.; LI, Weiping. **Applied nonlinear control**. New Jersey: Prentice Hall, 1991. Citado na página 12.

TAN, S.; HE, S. Hybrid control of nonlinear dynamical systems using neural nets and conventional control schemes. In: . [S.l.: s.n.], 1992. Citado na página 13.

TOKITA, M. et al. Position and force hybrid control of robotic manipulator by neural network (adaptive control of 2 dof manipulators). In: **[Proceedings] 1991 IEEE International Joint Conference on Neural Networks**. [S.l.: s.n.], 1991. p. 113–121 vol.1. Citado na página 13.

WANG, X.; XU, B.; DING, L. Simulation study on a single neuron pid control system of dc/dc converters. In: **2008 Workshop on Power Electronics and Intelligent Transportation System**. [S.l.: s.n.], 2008. p. 127–130. Citado na página 13.

WANG, Y.; LI, Y. On the temperature control of micro-fermentation tank based on single neuron pid algorithm. In: **2013 25th Chinese Control and Decision Conference (CCDC)**. [S.l.: s.n.], 2013. p. 436–439. Citado na página 13.

WERBOS, P. J. Neural networks for control and system identification. In: **Proceedings of the 28th IEEE Conference on Decision and Control,**. [S.l.: s.n.], 1989. p. 260–265 vol.1. Citado na página 13.

XIWEN, L. Single neuron self-tuning pid control for welding molten pool depth. In: **2008 7th World Congress on Intelligent Control and Automation**. [S.l.: s.n.], 2008. p. 7922–7925. Citado na página 18.

XU, B. et al. Neural control for longitudinal dynamics of hypersonic aircraft. In: **2013 International Conference on Unmanned Aircraft Systems (ICUAS)**. [S.l.: s.n.], 2013. p. 998–993. Citado na página 13.

ZILKOVA, Jaroslava; JAROSLAV, Timko; GIROVSKY, Peter. Nonlinear system control using neural networks. **Acta Polytechnica Hungarica**, v. 3, 10 2006. Citado na página 12.

Apêndices

APÊNDICE A – CÓDIGOS-FONTE DA SIMULAÇÃO DO PID NEURAL

A.1 PID NEURAL APLICADO AO CIRCUITO RC

```
clear all
close all

%Inicialização dos parâmetros
T=0.1; % tempo de amostragem
N=100; % número de iterações
Amax=5;
y=zeros(1,N); % saída da planta
u=zeros(1,N); % sinal de controle
e=zeros(1,N); % erro
w1=ones(1,N); % peso proporcional
w2=ones(1,N); % peso integral
w3=ones(1,N); % peso derivativo
I=zeros(1,N);
phiI=zeros(1,N);
deltaw1=zeros(1,N);
deltaw2=zeros(1,N);
deltaw3=zeros(1,N);
w1linha=zeros(1,N);
w2linha=zeros(1,N);
w3linha=zeros(1,N);

% Parâmetros de ajuste da parte P
eta1=0.1;
w1=w1*1.5;

% Parâmetros de ajuste da parte I
eta2=0.1;
w2=w2*0.1;

% Parâmetros de ajuste a parte D
eta3=0.01;
w3=w3*0.0;

% Função de transferência do circuito RC
```

```
R=10e3;
C=47e-6;
tau=R*C;
num=1;
den=[tau 1];
Gp=tf(num,den);
Gpd=c2d(Gp,T,'zoh'); %A partir da Gpd obtenho a eq. de dif.

% PID baseado em uma Rede Neural Artificial
for k=3:N
r(k)=1.0; % sinal de referência
y(k)=0.1917*u(k-1)+0.8083*y(k-1); % eq. de dif. da planta
e(k)=r(k)-y(k); % sinal do erro

x1(k)=e(k)-e(k-1);
x2(k)=e(k);
x3(k)=e(k)-2*e(k-1)+e(k-2);

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
u(k)=u(k-1)+phiI(k);

deltaw1(k)=eta1*x1(k)*e(k)*u(k);
deltaw2(k)=eta2*x2(k)*e(k)*u(k);
deltaw3(k)=eta3*x3(k)*e(k)*u(k);

w1linha(k)=w1(k-1)+deltaw1(k);
w2linha(k)=w2(k-1)+deltaw2(k);
w3linha(k)=w3(k-1)+deltaw3(k);

w1(k)=w1linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kp
w2(k)=w2linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Ki
w3(k)=w3linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kd

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
u(k)=u(k-1)+phiI(k);
end
```

```
% Plotar gráficos
k=0:N-1;

figure(1)
plot(k,r,'-.',k,y,'-')
legend('r(k)','y(k)');
title ('Saída do sistema com PID neural');
xlabel('kT');
ylabel('Saída y(k)');
grid on;

figure(2)
plot(k,u,'-');
legend('u(k)');
title ('Ação de controle do sistema com PID neural');
xlabel('kT');
ylabel('Sinal de controle u(k)');
grid on;

figure(3)
plot(k,w1,k,w2,k,w3)
legend('w1','w2','w3');
title ('Convergência dos pesos')
xlabel('kT');
ylabel('Amplitude');
grid on;

u=[zeros(1,2) ones(1,N-2)];
r=u;

y(1)=0.0;
y(2)=0.0;

for k=3:N
y(k)=0.1917*u(k-1)+0.8083*y(k-1);
end

k=0:N-1;
figure(4)
```

```
plot(k,r,'-.',k,y,'-');
legend('r(k)','y(k)');
title ('Saída do sistema em malha aberta');
xlabel('kT');
ylabel('y(k)');
grid on;
```

A.2 PID NEURAL APLICADO AO SISTEMA TORCIONAL

```
clear all
close all

%Inicialização dos parâmetros
T=0.009; % tempo de amostragem
N=250; % número de iterações
Amax=5;
y=zeros(1,N); % saída da planta
u=zeros(1,N); % sinal de controle
e=zeros(1,N); % erro
w1=ones(1,N); % peso proporcional
w2=ones(1,N); % peso integral
w3=ones(1,N); % peso derivativo
I=zeros(1,N);
phi1=zeros(1,N);
deltaw1=zeros(1,N);
deltaw2=zeros(1,N);
deltaw3=zeros(1,N);
w1linha=zeros(1,N);
w2linha=zeros(1,N);
w3linha=zeros(1,N);

% Parâmetros de ajuste da parte P
eta1=0.009;
w1=w1*0.0009;

% Parâmetros de ajuste da parte I
eta2=0.00001;
w2=w2*0.0001;
```

```
% Parâmetros de ajuste a parte D
eta3=0.09;
w3=w3*0.01;

% Função de transferência do sistema torcional
num=1600;
den=[1 0.83 0];
Gp=tf(num,den);
Gpd=c2d(Gp,T,'zoh'); % A partir de Gpd obtenho a equação de diferenças

% PID baseado em uma Rede Neural Artificial
for k=3:N
r(k)=1.0; % sinal de referência
% eq. de dif. da planta
y(k)=0.06464*u(k-1)+0.06448*u(k-2)+1.993*y(k-1)-0.9926*y(k-2);
e(k)=r(k)-y(k); % sinal do erro

x1(k)=e(k)-e(k-1);
x2(k)=e(k);
x3(k)=e(k)-2*e(k-1)+e(k-2);

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
u(k)=u(k-1)+phiI(k);

deltaw1(k)=eta1*x1(k)*e(k)*u(k);
deltaw2(k)=eta2*x2(k)*e(k)*u(k);
deltaw3(k)=eta3*x3(k)*e(k)*u(k);

w1linha(k)=w1(k-1)+deltaw1(k);
w2linha(k)=w2(k-1)+deltaw2(k);
w3linha(k)=w3(k-1)+deltaw3(k);

w1(k)=w1linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kp
w2(k)=w2linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Ki
w3(k)=w3linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kd

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
```

```
u(k)=u(k-1)+phil(k);
end

% Plotar gráficos
k=0:N-1;

figure(1)
plot(k,r,'-.',k,y,'-')
legend('r(k)','y(k)');
title ('Saída do sistema com PID neural');
xlabel('kT');
ylabel('Saída y(k)');
grid on;

figure(2)
plot(k,u,'-');
legend('u(k)');
title ('Ação de controle do sistema com PID neural');
xlabel('kT');
ylabel('Sinal de controle u(k)');
grid on;

figure(3)
plot(k,w1,k,w2,k,w3)
legend('w1','w2','w3');
title ('Convergência dos pesos')
xlabel('kT');
ylabel('Amplitude');
grid on;

u=[zeros(1,2) ones(1,N-2)];
r=u;

y(1)=0.0;
y(2)=0.0;

for k=3:N
y(k)=0.06464*u(k-1)+0.06448*u(k-2)+1.993*y(k-1)-0.9926*y(k-2);
end
```

```
k=0:N-1;
figure(4)
plot(k,r,'-.',k,y,'-');
legend('r(k)','y(k)');
title ('Saída do sistema em malha aberta');
xlabel('kT');
ylabel('y(k)');
grid on;
```

A.3 PID NEURAL APLICADO AO SISTEMA DE CONTROLE DE NÍVEL

```
clear all
close all

%Inicialização dos parâmetros
N=250; % número de iterações
Amax=1;
y=zeros(1,N); % saída da planta
u=zeros(1,N); % sinal de controle
e=zeros(1,N); % erro
w1=ones(1,N); % peso proporcional
w2=ones(1,N); % peso integral
w3=ones(1,N); % peso derivativo
I=zeros(1,N);
phiI=zeros(1,N);
deltaw1=zeros(1,N);
deltaw2=zeros(1,N);
deltaw3=zeros(1,N);
w1linha=zeros(1,N);
w2linha=zeros(1,N);
w3linha=zeros(1,N);

% Parâmetros de ajuste da parte P
eta1=0.1;
w1=w1*5.7;

% Parâmetros de ajuste da parte I
eta2=0.1;
```

```
w2=w2*0.1;

% Parâmetros de ajuste a parte D
eta3=0.1;
w3=w3*0.1;

% PID baseado em uma Rede Neural Artificial
for k=3:N
r(k)=1.0; % sinal de referência
% eq. de dif. da planta
y(k)=0.9722*y(k-1)+0.3578*u(k-1)
-0.1295*u(k-2)+0.3084*y(k-1)*y(k-2)*u(k-2)
-0.04228*y(k-2)^2+0.1633*y(k-2)*u(k-2)
-0.03259*y(k-2)*y(k-1)^2-0.3513*u(k-2)*y(k-1)^2
-0.3103*y(k-1)*u(k-1)+0.1087*y(k-2)*u(k-1)*u(k-2);
e(k)=r(k)-y(k); % sinal do erro

x1(k)=e(k)-e(k-1);
x2(k)=e(k);
x3(k)=e(k)-2*e(k-1)+e(k-2);

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
u(k)=u(k-1)+phiI(k);

deltaw1(k)=eta1*x1(k)*e(k)*u(k);
deltaw2(k)=eta2*x2(k)*e(k)*u(k);
deltaw3(k)=eta3*x3(k)*e(k)*u(k);

w1linha(k)=w1(k-1)+deltaw1(k);
w2linha(k)=w2(k-1)+deltaw2(k);
w3linha(k)=w3(k-1)+deltaw3(k);

w1(k)=w1linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kp
w2(k)=w2linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Ki
w3(k)=w3linha(k)/sqrt(w1linha(k)^2+w2linha(k)^2+w3linha(k)^2); % Kd

I(k)=x1(k)*w1(k)+x2(k)*w2(k)+x3(k)*w3(k);
phiI(k)=(Amax*(1-exp(-I(k))))/(1+exp(-I(k)));
```

```
u(k)=u(k-1)+phil(k);
end

% Plotar gráficos
k=0:N-1;

figure(1)
plot(k,r,'-.',k,y,'-')
legend('r(k)','y(k)');
title ('Saída do sistema com PID neural');
xlabel('kT');
ylabel('Saída y(k)');
grid on;

figure(2)
plot(k,u,'-');
legend('u(k)');
title ('Ação de controle do sistema com PID neural');
xlabel('kT');
ylabel('Sinal de controle u(k)');
grid on;

figure(3)
plot(k,w1,k,w2,k,w3)
legend('w1','w2','w3');
title ('Convergência dos pesos')
xlabel('kT');
ylabel('Amplitude');
grid on;

u=[0*ones(1,2) 1*ones(1,N-2)];
r=u;
N=length(u);

y(1)=0.0;
y(2)=0.0;

for k=3:N
y(k)=0.9722*y(k-1)+0.3578*u(k-1)
```

```
-0.1295*u(k-2)+0.3084*y(k-1)*y(k-2)*u(k-2)
-0.04228*y(k-2)^2+0.1633*y(k-2)*u(k-2)
-0.03259*y(k-2)*y(k-1)^2-0.3513*u(k-2)*y(k-1)^2
-0.3103*y(k-1)*u(k-1)+0.1087*y(k-2)*u(k-1)*u(k-2);
end

k=0:N-1;
figure(4)
plot(k,r,'-.',k,y,'-');
legend('r(k)','y(k)');
title ('Saída do sistema em malha aberta');
xlabel('kT');
ylabel('Saída y(k)');
grid on;
```